Effective pre-scales for DSM based minbias trigger

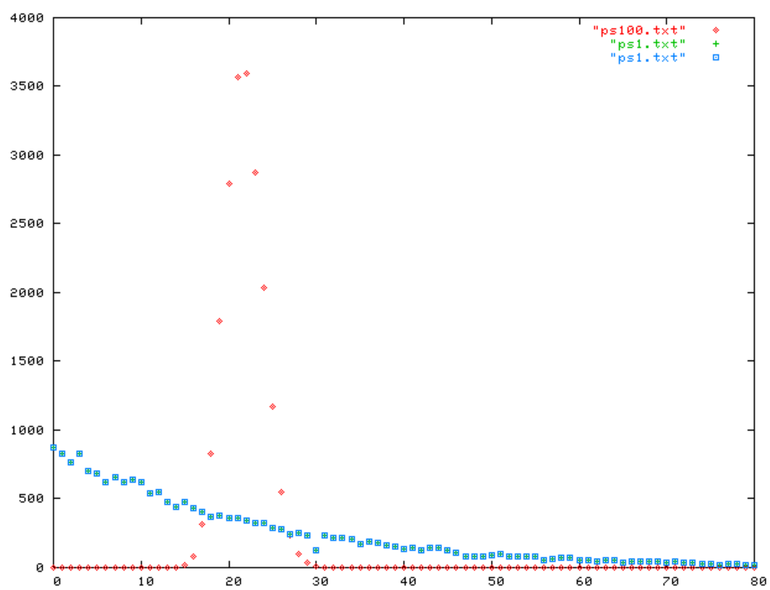
Carl pointed out a interpretation difficulty, regarding the DSM prescaled minbias trigger. He raised it as a "trigger dependent deadtime."

This description is based on the formula:

Npresented = Nrecorded * ps * deadtime

Which doesn't hold for the scaled minbias if you insist that ps = DSM_ps * TCU_ps. I prefer to keep "deadtime" to mean the true % time the detector is dead. The problem is with the assertion that ps = DSM_ps * TCU_ps and so I will analyze the problem in terms of "effective prescales." The words you use don't matter, its only a matter of which closet you choose to sweep the mess into.

Either way you describe it, the issue comes up because the arrival time distribution prescaled events is different than that of prescaled events:



The blue is the arrival time histogram, in bunch crossings, for unprescaled minbias events at 450khz input rate. The red is the same for events prescaled by 100 (the x-axis is compressed by the prescale value.)

Carl noted that unlike the standard TCU prescale, the DSM prescale is applied before detector deadtime. To see the problem, pretend the TPC deadtime is very short (<10,000 bunch crossings.) If so, the scaled mb event never arrives before ~20,000 bunch crossings. When it does finally arrive, the TPC will always be live.

The formula:

Nevents_to_tape = livetime * scaler_rate * ps

Does not apply. (here you see why Carl uses "trigger dependent deadtime" – if you set livetime = 1, the formula works)

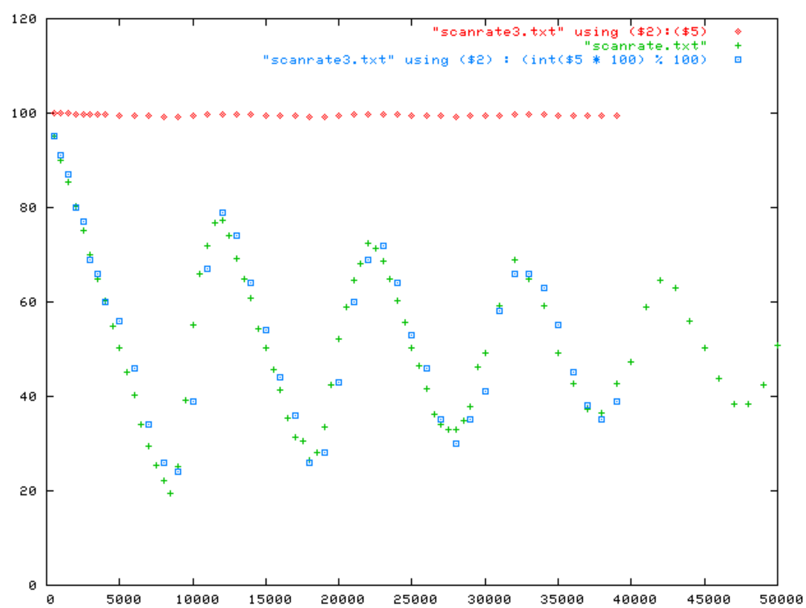
The current situation, for the pp run is more complicated. In this situation, we have a TCU prescale, in addition to the DSM prescale. The TCU prescale is applied after the detector deadtime.

You can calculate what the effective prescale should be:

ps_effective = (TCU_ps-1) * DSM_ps + DSM_ps/2

This assumes that the time from the moment the TPC goes live till the first scaled MB bit arrives is randomized, but must be DSM_ps/2 min bias events on average. This is the point where the TCU_ps is first decremented. However, from this point the expectation value for the number of live bunch crossings till the next scaled MB bit is just the standard DSM_ps MB events, whether or not the detector goes dead in the mean time.

This is a statistical argument that assumes that the dead time is long with respect to the arrival of scaled DSM bits. I did a simulation to figure out the actual behavior:



Here, the x axis is the minbias rate in hz. The y axis is the effective prescale divided by the real prescale. If the standard definition for the prescale holds (ps = DSMps * TCUps) this value should be a constant 100 (The DSM prescale).

The specific definitions of these quantities are:

> "Effective prescale" = livetime * scaler_counts / recorded_counts
> "real prescale" = DSM_ps * TCU_ps

All lines are scaled to the TPC's dead time and the current DSM prescale of 100. The red line is for a TCU prescale of 100. The green line uses a TCU prescale set to 1. The blue line effective ps – 99*100, for the TCU prescale of 100. It is just a expanded view of the oscillation in the red line and shows the reason is the same edge effect that causes the oscillation when TCU prescale = 1.

The oscillation appears because as you scan the rate you go through phases where the scaled bit turns on just before, or just after the TPC goes live again. At a very low event rate, the TPC deadtime compared to the number of cycles for an event to arrive As the rate increases, the deadtime increases, even though the detector is always live when the events arive, so the effective prescale goes down. But at a mb rate ~8000hz, the events start arriving before the detector is live, so the effective prescale rises again. The oscillation continues, the value raising when the mean arrival time of a scaled minbias event lands during the time the detector is dead, and decreases when it lands during the time the detector is live.

The relevant regimes:

For the current pp run, the effective prescale differs little from the calculated DSM_ps * TCU_ps, because the value for TCU_ps is large (~10000), so the relative variation is ~<1/10000.

For the 62GeV run, assuming the following parameters:

> 5khz minbias rate
> 25 hz desired rate to tape

The TCU prescale will have to be 1 or 2. As Carl pointed out, this does put us in the regime where the effective prescales are problematic.

Can this effect bias the trigger?

The requirement for ensuring that a trigger is unbiased is that the relative representation of each contributing trigger word is the same in the data file as it is in the presented events.

All triggers either contain the scaled MB bit requirement for all contributing trigger words or else don't require it at all, therefore, I don't think its possible for this effect to bias our triggers.

However, if we set up a mixed trigger:

Condition = “scaled_MB and X” or “Y”

For example, we could get situations where the trigger words related to the condition “scaled_MB & X” part was suppressed relative to the “Y” condition.

In practice, I don’t see why such a trigger would be made. The closest we have is our L2 based zerobias trigger defined as: “minbias & JPsi” or “zerobias”, but this trigger would be immune for two reasons:

1. It not based on the scaled minbias.
2. The L2 algorithm reapplies the TCU algorithm and separates each into different components anyway.

What Should we do?

We have two options. The first is to avoid the issue entirely by not using the prescaled minbias during the 62GeV run. The repercussion of this is that we must give up the fpd1_fast_tpclive trigger.

The second option is to do nothing. The only action item would to remind ourselves that we need to be very careful obtaining luminosities. In the 62 GeV run, the current scheme of calculating the sampled luminosity according to number of recorded MB events will not work. This is based on the calculation:

Sampled = (Npresented / sigma) * (deadtime)

sigma = 25mb / DSMps // the scale reduces the xsection by 100
deadtime = Nrecorded * TCU_ps / Npresented

so,

Sampled = Nrecorded * TCUps * DSMps / 25mb

But this deadtime calculation will no longer valid. Instead we need to go to the scalers:

S = Nrecorded / sigma

sigma = sigma(scaled_mb) * (Npresented / Npresented(scaled_mb))

so,

S = (100/25mb) * Nrecorded * (Npresented(scaled_mb)/Npresented)

Where the presented values are taken from the scalers, and has no dependence on detector deadtimes nor on prescale values.

The same care needs to be taken analyzing cross sections offline. In particular we need to avoid 2 practices for the scaled minbias data:

1. Don't assume Npresented = TCU_ps * DSM_ps * Nrecorded
2. Because of #1, don't assume ratios between scaled mb and any other trigger has any clear relationship to the cross section.